

SAE : Linux

Partie 1

Groupe B2 GEII 3^{ème} année
IUT de l'Aisne, Cuffies, 2024

Sommaire :

- 1/ Création d'un groupe
- 2/ Création d'un utilisateur
- 3/ Travail sur papier
- 4/ La commande ls
- 5/ Travail sur papier
- 6/ La commande cd
- 7/ Les commandes mkdir et rmdir
- 8/ Travail sur machine
- 9/ Lien et mode d'un fichier
- 10/ Sticky BIT
- 11/ Les ACL Unix
- 12/ Les redirections
- 13/ Les tubes et les filtres
- 14/ L'interpréteur de la commande Bash
- 15/ Les variables locales
- 16/ Les variables d'environnement
- 17/ Les substitution de commandes
- 18/ Code de retour d'une commande
- 19/ Les métacaractères
- 20/ Les processus Unix
- 21/ Mise en pause et reprise d'un processus
- 22/ Les scripts-shell
- 23/ Conclusion

1/ Création d'un groupe

A l'aide de la commande `addgroup`, créer un groupe `developpeurs` avec un GID égal à 600

```
~/Bureau$ sudo groupadd -g 600 developpeurs
```

2/ Création d'un utilisateur

A l'aide de la commande `adduser`, créer un compte nommé "raoul" pour l'utilisateur Raoul, d'UID égal à 601, appartenant au groupe `developpeurs`, et dont le répertoire de connexion est `/home/raoul`

```
paternotte@paternotte-VirtualBox:~/Bureau$ sudo useradd -m raoul
```

```
paternotte@paternotte-VirtualBox:~/Bureau$ id raoul
uid=601(raoul) gid=1002(raoul) groupes=1002(raoul)
```

```
paternotte@paternotte-VirtualBox:~/Bureau$ echo ~raoul
/home/raoul
```

```
paternotte@paternotte-VirtualBox:~/Bureau$ sudo usermod -a -G developpeurs raoul
```

```
paternotte@paternotte-VirtualBox:~/Bureau$ groups raoul
raoul : raoul developpeurs
```

A partir de la session courante, se connecter en tant que Raoul. Vérifier avec `id`. Constater que l'uid et le GID de Raoul sont corrects. Se positionner dans le répertoire de connexion. Vérifier avec `pwd`. Avec la commande `exit`, fermer la session.

```
paternotte@paternotte-VirtualBox:~/Bureau$ su raoul
Mot de passe :
$ id
uid=601(raoul) gid=1002(raoul) groupes=1002(raoul),600(developpeurs)
$ exit
```

Avec la commande `id` retrouver vos informations et `getent passwd "nom"`

```
paternotte@paternotte-VirtualBox:~/Bureau$ getent passwd paternotte
paternotte:x:1000:1000:paternotte:/home/paternotte:/bin/bash
```

```
paternotte:x:1000:1000:paternotte:/home/paternotte:/bin/bash
```

Utilisateur: paternotte

UID:1000, GID:1000

Commentaire: paternotte

répertoire personnel : /home/paternotte

shell: /bin/bash

Avec getent group "nom" retrouver les information du groupes

```
paternotte@paternotte-VirtualBox:~/Bureau$ getent group developpeurs
developpeurs:x:600:raoul
```

group: developpeurs
GID: 600

personne : raoul

Avec id root trouver les information du super utilisateur

```
paternotte@paternotte-VirtualBox:~/Bureau$ id root
uid=0(root) gid=0(root) groupes=0(root)
```

UID=0 GID=0

3/ Travail sur papier

\$ Cat /etc/group

compta:x:100:Pierre,Julie

commerce:x:200:Julie,Ernestine,Raoul

direction:x:500:Fernand

\$ Cat /etc/passwd

Pierre:x:101:100:Pierre:/home/compta/pierre:/bin/bash

Julie:x:102:100:Julie:/home/compta/julie:/bin/bash

Fernand:501:500:Fernand:/home/fernand:/bin/bash

Ernestine:x:201:200:Ernestine:/home/comm/Ernestine:/bi/bash

Raoul:x:202:200:Raoul:/home/comm/Raoul:/bin/ksh

1/ Donner l'UID de l'utilisateur Raoul

L'UID de l'utilisateur Raoul est 202

2/ Donner le GID du groupe principal de l'utilisatrice Julie

Le GID du groupe principal de l'utilisatrice Julie est 100

3/ Donner le nom de l'éventuel groupe supplémentaire de l'utilisatrice Julie

Le nom de l'éventuel groupe supplémentaire de l'utilisatrice Julie est commerce de GID 200

4/ Quel est le login de l'utilisateur dont l'UID est égal à 501 ?

Le login de l'utilisateur dont l'UID est égal à 501 est fernand

5/ Pierre fait-il partie d'un groupe supplémentaire ? Si oui, lequel ?

Pierre fait seulement partie du groupe compta de GID 100 et n'a pas de groupe supplémentaire

4/La commande ls

Sans argument, ls affiche le contenu du répertoire courant. Essayer et Que s'affiche-t-il?

```
paternotte@paternotte-VirtualBox:~$ ls
adduser.conf      fuse.conf         Musique           sudo.conf
apg.conf          gai.conf          nftables.conf    sudo_logsrvd.conf
beurk             hdparm.conf      nsswitch.conf    sysctl.conf
brltty.conf       host.conf         OK               Téléchargements
Bureau            Images            pam.conf         ucf.conf
ca-certificates.conf kerneloops.conf  pasOK            usb_modeswitch.conf
coucouter         ld.so.conf        pnm2ppa.conf     vconsole.conf
debconf.conf      lettres           pre              Vidéos
deluser.conf      libao.conf        Public            voiture
dhcpcd.conf       libaudit.conf     resolv.conf      voitures
Documents         locale.conf       rsyslog.conf     xattr.conf
e2scrub.conf      logrotate.conf    rygel.conf
err               mke2fs.conf       sensors3.conf
fprindtd.conf     Modèles           snap
```

La commande ls permet d'afficher le contenu du répertoire courant, listant ainsi les fichiers et dossiers présents dans le répertoire où la commande est exécutée.

Si on fournit en argument un ou des répertoires, ls affiche le contenu de ces répertoires. Essayer la commande ls /bin. Résultat ? « /bin » est-il un chemin absolu ou relatif ?

```
paternotte@paternotte-VirtualBox:/home$ ls /bin
['
aa-enabled        nc
aa-exec           ncal
aa-features-abi  nc.openbsd
aconect           neqn
acpidbg           netaddr
add-apt-repository netcat
addpart           networkctl
airscan-discover networkd-dispatcher
alsabat           newgrp
alsaloop          ngettext
alsamixer         nice
alsatplg          nisdomainname
alsaucm           nl
amidi             nm-applet
amixer            nmcli
apg               nm-connection-editor
apgbfm            nm-online
aplay             nmtui
aplaymidi         nmtui-connect
apport-bug        nmtui-edit
apport-cli        nmtui-hostname
```

La commande ls /bin permet d'afficher le contenu du répertoire /bin, où sont stockés les fichiers exécutables essentiels au fonctionnement du système. Le chemin /bin est un chemin absolu : les chemins absolus commencent toujours par une barre oblique (/) et sont définis à partir de la racine du système de fichiers.

Essayer la commande `ls /home /var/spool`. Résultat ?

```
paternotte@paternotte-VirtualBox:~$ ls /home /var/spool
/home:
paf  paternotte  pp  raoul

/var/spool:
anacron  cron  cups  libreoffice  mail  rsyslog
```

La commande `ls /home /var/spool` affiche le contenu des répertoires `/home` et `/var/spool`, montrant ainsi les fichiers et dossiers présents dans chacun de ces emplacements.

Remarquer que la commande ci-dessus admet deux arguments. En effet, la commande `ls` peut accepter un nombre quelconque d'arguments.

La commande `ls` admet beaucoup d'options, parmi lesquelles :

-Option `-a` : affiche les fichiers cachés. Sous Unix, un fichier est caché si son nom commence par un point « . ». C'est un fichier tout à fait normal ; sa seule particularité est qu'il n'apparaît pas quand on utilise la commande `ls` sans l'option `-a`.

Étant dans votre répertoire personnel, essayez les commandes `ls` puis `ls -a`. Que remarquez-vous ?

```
paternotte@paternotte-VirtualBox:~$ ls -a
.          gai.conf          pnm2ppa.conf
..         .gnupg             pre
adduser.conf  hdpam.conf        .profile
apg.conf     host.conf         Public
.bash_history Images            resolv.conf
.bash_logout kerneloops.conf  rsyslog.conf
.bashrc      ld.so.conf       rygel.conf
beurk        .lessht          sensors3.conf
brltty.conf  lettres          snap
Bureau       libao.conf       .ssh
ca-certificates.conf libaudit.conf   .sudo_as_admin_successful
.cache       .local           sudo.conf
.config     locale.conf     sudo_logsrvd.conf
coucouter   logrotate.conf  sysctl.conf
debconf.conf mke2fs.conf     Téléchargements
deluser.conf Modèles         ucf.conf
dhcpcd.conf Musique        usb_modeswitch.conf
Documents   nftables.conf   vconsole.conf
e2scrub.conf nsswitch.conf   Vidéos
err         OK              voiture
fprintd.conf pam.conf        voitures
fuse.conf   pasOK          xattr.conf
```

La commande `ls -a` permet également d'afficher les fichiers cachés, qui ont un nom commençant par un point (`. `).

Option -i : affichage long. Affiche toutes les caractéristiques des fichiers, c'est-à-dire l'intégralité du i-nœud. Essayer ls -i. Que remarquez-vous ?

```
paternotte@paternotte-VirtualBox:~$ ls -i
394494 adduser.conf      393383 Images             393380 Public
394515 apg.conf           394528 kerneloops.conf     394539 resolv.conf
393493 beurk               394529 ld.so.conf           394540 rsyslog.conf
394516 brltty.conf          394469 lettres              394541 rygel.conf
393377 Bureau                394530 libao.conf           394542 sensors3.conf
394518 ca-certificates.conf 394531 libaudit.conf        393241 snap
393398 coucouter             394532 locale.conf         394543 sudo.conf
394519 debconf.conf          394533 logrotate.conf      394544 sudo_logsrvd.conf
394520 deluser.conf          394534 mke2fs.conf          394545 sysctl.conf
394521 dhcpd.conf            393379 Modèles              393378 Téléchargements
393381 Documents              393382 Musique              394546 ucf.conf
394522 e2scrub.conf           394535 nftables.conf        394547 usb_modeswitch.conf
394484 err                    394536 nsswitch.conf        394548 vconsole.conf
394523 fprintd.conf           394461 OK                   393384 Vidéos
394524 fuse.conf             394537 pam.conf            394489 voiture
394525 gai.conf              394482 pasOK                394487 voitures
394526 hdparm.conf           394538 pnm2ppa.conf         394549 xattr.conf
394527 host.conf             394481 pre
```

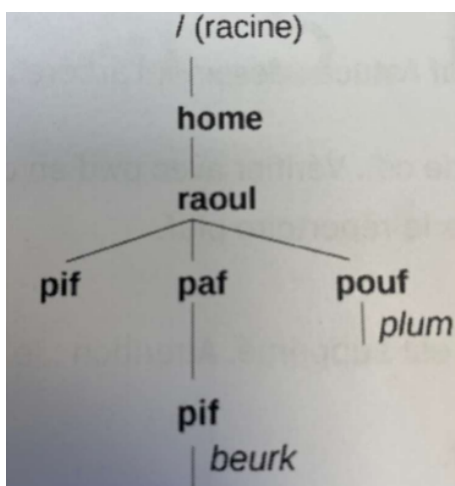
La commande `ls -i` permet d'afficher les inodes des fichiers. Chaque fichier dans un système de fichiers Unix possède un inode unique, qui sera affiché aux côtés d'autres informations sur les fichiers. Un inode (ou *nœud d'index*, contraction des termes anglais *index* et *node*) est une structure de données contenant des informations sur un fichier ou un répertoire, propre à certains systèmes de fichiers, notamment sous Linux et Unix.

5/ Travail sur papier

Soit l'arborescence suivante :

Les fichiers de type répertoire sont indiqués en gras

Les fichiers réguliers (aussi appelés « ordinaires ») sont indiqués en italique



1.1/ On suppose que le répertoire courant est /home/raoul.

2.1/ Que vaut « .. » ?

Le répertoire parent de /home/raoul est /home, donc « .. » vaut /home. « .. » représente le répertoire parent.

3.1/ Donner un chemin d'accès absolu au fichier beurk.

Le chemin absolu pour le fichier beurk est : cd /home/raoul/paf/pif/beurk.

4.1/ Donner un chemin d'accès relatif au fichier plum

Le chemin relatif pour le fichier plum depuis /home/raoul est : cd pouf/plum.

5.1/ Quel répertoire ne possède pas de répertoire parent?

Si « / (racine) » est considéré comme un répertoire alors c'est « / (racine) » qui n'a pas de répertoire parent sinon c'est le répertoire home qui n'a pas de répertoire parent.

On suppose maintenant que le répertoire courant devient /home/raoul/paf 1.

2/ Que vaut « . »

« . » représente le répertoire courant, donc /home/raoul/paf.

2.2/ Donner un chemin d'accès relatif au fichier /home/raoul/pif

Le chemin relatif pour le fichier /home/raoul/pif depuis /home/raoul/paf est : cd ../pif.

3.2/ Donner un chemin d'accès relatif au fichier plum

Le chemin relatif pour le fichier /home/raoul/pouf/plum depuis /home/raoul/paf est :
cd ../pouf/plum.

On suppose maintenant que le répertoire courant devient /home/raoul/paf/pif

1.3/ Que vaut « ../.. » ?

« ../.. » représente le répertoire deux niveaux au-dessus du répertoire courant, donc /home/raoul.

2.3/ Donner un chemin d'accès relatif au fichier/home/raoul/pif.

Le chemin relatif pour le fichier /home/raoul/pif depuis /home/raoul/paf/pif est : cd
../../pif.

6/ La commande cd

cd (change directory) : naviguer dans l'arborescence.

Admet en argument le chemin (absolu ou relatif) du catalogue dans lequel on souhaite se positionner.

Essayer la commande cd /usr ; « /usr » est il un chemin absolu ou relatif ?
Vérifier le résultat avec la commande pwd.

```
paternotte@paternotte-VirtualBox:/home$ cd /usr
paternotte@paternotte-VirtualBox:/usr$ pwd
/usr
```

En utilisant la commande cd /usr, j'ai constaté que « /usr » est un chemin absolu. La commande pwd a confirmé que je me trouvais bien dans le répertoire « /usr ».

Essayer la commande cd local. « local » est il un chemin absolu ou relatif ?
Vérifier le résultat avec la commande pwd. Quel est donc le chemin absolu du répertoire courant ?

```
paternotte@paternotte-VirtualBox:/usr$ cd local
paternotte@paternotte-VirtualBox:/usr/local$ pwd
/usr/local
```

La commande `cd local` utilise un chemin relatif, et j'ai exécuté `pwd` pour vérifier le résultat et obtenir le chemin absolu du répertoire courant, qui est « /usr/local ».

Essayer la commande `cd ..` : vous accédez au répertoire parent du répertoire courant, c'est-à-dire que vous «remontez d'un cran» dans l'arborescence. Vérifier le résultat avec la commande `pwd`. Quel est donc le répertoire courant ?

```
paternotte@paternotte-VirtualBox:/usr/local$ cd ..
paternotte@paternotte-VirtualBox:/usr$ pwd
/usr
```

En utilisant la commande ``cd ..``, on remonte d'un niveau dans l'arborescence. J'ai vérifié le résultat avec ``pwd`` pour confirmer le répertoire courant après cette opération, qui est « /usr ».

Essayer la commande `cd`. Quel est maintenant le répertoire courant ? Conclure sur le rôle de la commande `cd` utilisée sans argument ?

```
paternotte@paternotte-VirtualBox:/usr$ cd
paternotte@paternotte-VirtualBox:~$
```

En exécutant la commande `cd` sans argument, puis en vérifiant le répertoire courant avec `pwd`, on constate qu'on est renvoyé vers le répertoire personnel, qui devient alors le répertoire courant. En conclusion, la commande `cd` sans argument ramène au répertoire personnel, tandis que l'utilisation de `cd` avec un chemin (absolu ou relatif) permet de naviguer vers un autre répertoire.

7/ Les commandes `mkdir` et `rmdir`

Essayez la commande `mkdir plouf` : vous créez un répertoire `plouf` dans votre répertoire courant. Vérifiez avec la commande `ls -l` et remarquez le petit « d » en début de ligne qui indique que `plouf` est bien un répertoire (directory en anglais).

```
paternotte@paternotte-VirtualBox:~$ mkdir plouf

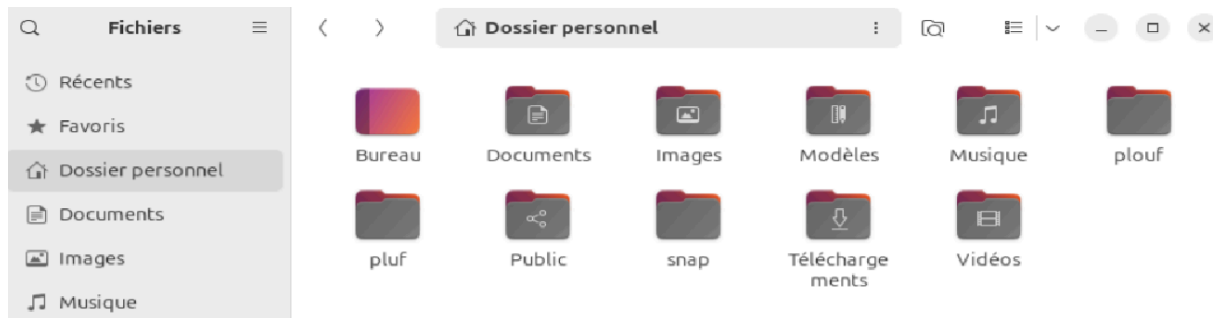
paternotte@paternotte-VirtualBox:~$ ls -l
total 40
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Bureau
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Documents
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Images
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Modèles
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Musique
drwxrwxr-x 2 paternotte paternotte 4096 oct. 23 16:12 plouf
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Public
drwx----- 5 paternotte paternotte 4096 oct. 23 09:40 snap
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Téléchargements
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Vidéos
```

On peut observer un petit « d » au début de la ligne, ce qui confirme que « `plouf` » est un répertoire.

Essayer la commande `mkdir pluf plouf/plaf` : vous créez deux répertoires (`mkdir` accepte un nombre quelconque d'arguments) :

- Un dénommé `pluf` dans le répertoire courant
- Un autre dénommé `plaf` dans le répertoire `plouf`. Astuce : dessiner l'arborescence obtenue sur un papier

```
paternotte@paternotte-VirtualBox:~$ mkdir plouf
paternotte@paternotte-VirtualBox:~$ mkdir pluf
paternotte@paternotte-VirtualBox:~$ mkdir plouf/plaf
```



Faites de `plaf` votre répertoire courant (commande `cd`). Vérifier avec `pwd` en cas de doute,

```
paternotte@paternotte-VirtualBox:~$ cd plouf
paternotte@paternotte-VirtualBox:~/plouf$ cd plaf
paternotte@paternotte-VirtualBox:~/plouf/plaf$ pwd
/home/paternotte/plouf/plaf
```

En restant dans `plaf`, créer un répertoire `plif` dans le répertoire `pluf`.

```
paternotte@paternotte-VirtualBox:~/plouf/plaf$ mkdir /home/paternotte/pluf/plif
```

Faire du répertoire `plouf` le répertoire courant.

```
paternotte@paternotte-VirtualBox:~/plouf/plaf$ cd ..
```

Taper la commande `rmdir plaf` : le répertoire `plaf` est supprimé. Attention : le répertoire à supprimer doit être vide !

```
paternotte@paternotte-VirtualBox:~/plouf$ rmdir plaf
```

< > Dossier personnel / plouf

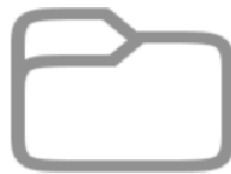


Le dossier est vide

En restant dans plouf, supprimer le répertoire plif.

```
paternotte@paternotte-VirtualBox:~/plouf$ rmdir /home/paternotte/plouf/plif
```

< > Dossier personnel / plouf



Le dossier est vide

8/ Travail sur machine

Créer dans le répertoire de connexion un répertoire pif

```
paternotte@paternotte-VirtualBox:~$ mkdir pif
```

Créer dans ce répertoire la sous-arborescence décrite ci-dessous en restant au niveau du répertoire pif

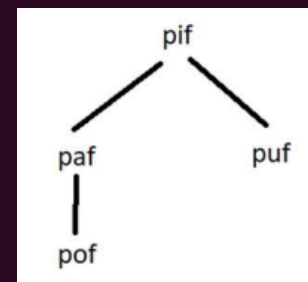
```
paternotte@paternotte-VirtualBox:~$ mkdir -p pif/paf/pof pif/puf
paternotte@paternotte-VirtualBox:~$ ls -R pif
```

```
pif:
paf  puf
```

```
pif/paf:
pof
```

```
pif/paf/pof:
```

```
pif/puf:
```



Se placer dans le sous-répertoire pof.

```
paternotte@paternotte-VirtualBox:~/pif$ cd paf/pof
paternotte@paternotte-VirtualBox:~/pif/paf/pof$
```

Copier dans ce répertoire le fichier /etc/group

```
paternotte@paternotte-VirtualBox:~/pif/paf/pof$ cp -r /etc/group /home/paternotte/pif/paf/pof
```



Revenir au niveau du répertoire pif et détruire le répertoire puf.

```
paternotte@paternotte-VirtualBox:~/pif$ rmdir puf
```

En restant dans pif, afficher à l'écran le contenu du fichier group situé dans pof.

```
paternotte@paternotte-VirtualBox:~/pif$ cat paf/pof/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,paternotte
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:paternotte
floppy:x:25:
tape:x:26:
```

Déplacer le fichier group dans le répertoire paf

```
~/pif$ mv paf/pof/group paf/
```



Supprimer ce fichier group.

```
~/pif$ rm paf/group
```

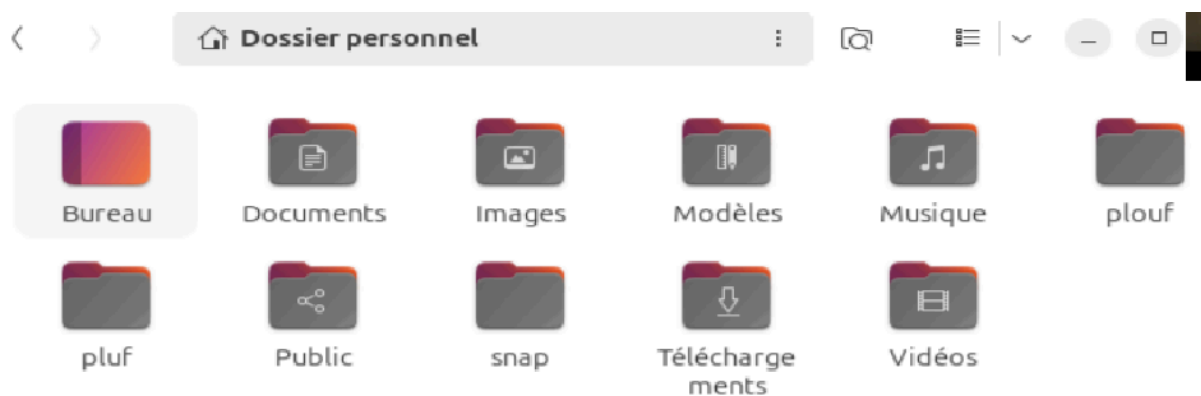


Se placer dans le répertoire parent de pif

```
~/pif$ cd ..  
~$
```

Finalement, supprimer le répertoire pif et tout ce qu'il contient (utiliser l'option appropriée de rm, consulter l'aide en ligne).

```
~$ rm -r pif
```



9/ Lien et mode d'un fichier

Cet exercice doit être traité par deux utilisateurs d'un même groupe « développeurs » qui travaillent chacun sous leur propre login

Structure des permissions

Les permissions sont généralement représentées par trois catégories :

1. User (u) : le propriétaire du fichier.
2. Group (g) : le groupe auquel appartient le fichier.
3. Other (o) : tous les autres utilisateurs.

Chaque catégorie peut avoir les permissions suivantes :

- r (read) : permission de lire le fichier (valeur 4).
- w (write) : permission d'écrire ou de modifier le fichier (valeur 2).
- x (execute) : permission d'exécuter le fichier s'il s'agit d'un programme ou d'entrer dans un répertoire (valeur 1).

Calcul des permissions

Pour définir les permissions, on additionne les valeurs correspondantes pour chaque catégorie :

- Par exemple, si un utilisateur a les droits de lecture et d'écriture (rw-), cela correspond à $4+2=6$.
- Si le groupe a seulement le droit de lecture (r--), cela correspond à 4.
- Si les autres n'ont aucun droit (---), cela correspond à 0.

Utilisateur A (paternotte)

Positionner les droits du répertoire de connexion à rwx r-x ---.

```
paternotte@paternotte-VirtualBox:~$ chmod 750 /home/paternotte
paternotte@paternotte-VirtualBox:~$ ls -ld
drwxr-x--- 19 paternotte paternotte 4096 nov.  4 03:12 .
```

Créer un catalogue nommé lettres puis utiliser la commande chmod pour établir ses droits d'accès à rwx rwx ---.

```
paternotte@paternotte-VirtualBox:~$ chmod 755 /home/paternotte
paternotte@paternotte-VirtualBox:~$ ls -l
total 36
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Bureau
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Documents
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Images
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Modèles
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Musique
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Public
drwx----- 5 paternotte paternotte 4096 oct. 23 09:40 snap
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Téléchargements
drwxr-xr-x 2 paternotte paternotte 4096 oct. 17 09:50 Vidéos
```

utilisateur B (Raoul)

Se positionner dans le répertoire lettres créé ci-dessus par l'utilisateur A
Créer avec un éditeur de texte un fichier nommé souvenir contenant la phrase
« vive Unix » Pourquoi l'utilisateur B a-t-il le droit de créer souvenir ?

```
paternotte@paternotte-VirtualBox:~$ sudo usermod -aG 1000 raoul
paternotte@paternotte-VirtualBox:~$ su raoul
Mot de passe :
$ cd ..
$ cd paternotte
$ cd lettres
$ ls -l
total 0
$ ls -ld
drwxrwx--- 2 paternotte paternotte 4096 oct. 25 09:37 .
$ nano souvenir
$ cat souvenir
vive Unix
```

L'utilisateur B a la possibilité de créer le fichier « souvenir » puisqu'il appartient au même groupe que l'utilisateur A. Il dispose de tous les droits dans le répertoire « lettres » (rwx rwx ---), avec les droits de l'utilisateur B indiqués en rouge.

Positionner les droits d'accès à ce fichier à rw- r-- ---

```
$ chmod u+rw souvenir
$ chmod u-x souvenir
$ chmod g+r souvenir
$ chmod g-wx souvenir
$ chmod o-rwx souvenir

$ ls -ld souvenir
-rw-r----- 1 raoul raoul 10 oct. 25 10:05 souvenir
```

Quels sont les droits de l'utilisateur A sur ce fichier souvenir ?

L'utilisateur A a seulement le droit de lecture (r), car il appartient au même groupe. Ses droits sont définis comme (rw- r-- ---), avec les droits de l'utilisateur A indiqués en rouge.

10/ Sticky BIT

Cette question doit être traitée par les mêmes utilisateurs que précédemment.
Utilisateur B. (raoul)

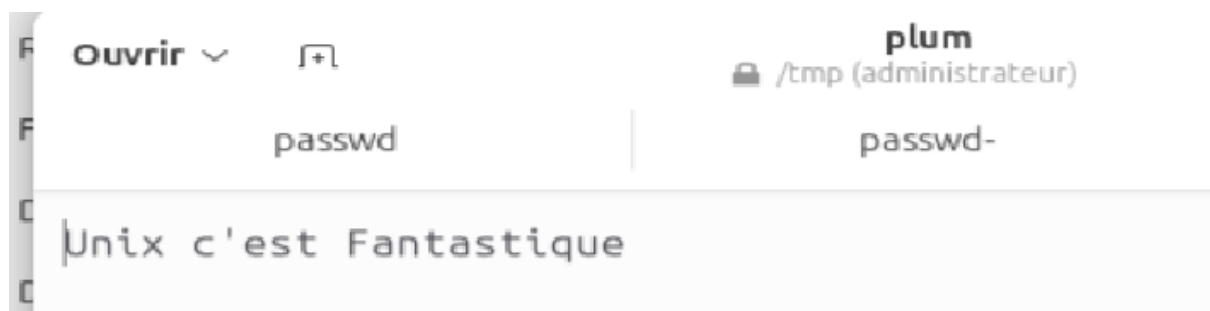
Examiner attentivement les droit d'accès du répertoire /tmp

```
drwxrwxrwt 19 root root 4096 oct. 25 09:58 tmp
```

Remarquer le petit « t » qui signifie le sticky bit est positionné sur ce répertoire.

Créer dans ce répertoire un fichier nommé plum contenant la phrase « Unix c'est fantastix »

```
$ nano plum
```



Positionner les droits d'accès à ce fichier à rw- r—r—

```
$ chmod 644 plum
$ ls -ld plum
-rw-r--r-- 1 raoul raoul 23 oct. 25 10:21 plum
```

Quels sont les droits de l'utilisateur A sur ce fichier ?

L'utilisateur A peut lire le fichier plum car il fait partie du même groupe que l'utilisateur B mais il ne peut pas le supprimer car le sticky bit l'empêche (t). (rw- r—r--), droit de l'utilisateur A sur ce fichier en rouge.

Utilisateur A. (Paternotte)

Afficher le contenu du fichier plum. Possible ? Pourquoi ?

```
paternotte@paternotte-VirtualBox:/tmp$ cat plum
Unix c'est fantastique
```

On peut accéder au fichier « plum » car il appartient au même groupe que l'utilisateur B, et l'utilisateur A a reçu le droit de lecture (rw- r—r--). Les droits de l'utilisateur A sur ce fichier sont indiqués en rouge.

Rappeler les droits de l'utilisateur A sur le répertoire /tmp

```
paternotte@paternotte-VirtualBox:/tmp$ ls -ld /tmp
drwxrwxrwt 19 root root 4096 nov. 4 03:42 /tmp
```


Détruire ce fichier. Possible ? Etrange, pas vrai ? Pourquoi ?

```
paternotte@paternotte-VirtualBox:/tmp$ rm plum
rm : supprimer 'plum' qui est protégé en écriture et est du type « fichier » ?

paternotte@paternotte-VirtualBox:/tmp$ rmdir plum
rmdir: impossible de supprimer 'plum': Opération non permise
```

Le sticky bit empêche la destruction de ce fichier (t).

Utilisateur B

Détruire ce fichier. Possible ? Pourquoi ?

```
$ rm plum
$ cat plum
cat: plum: Aucun fichier ou dossier de ce nom
```

Cela est possible car il est propriétaire et peut détruire le fichier malgré le sticky bit

11/ Les ACL Unix

Cet exercice doit être traité par deux utilisateurs d'un même groupe principal.

Utilisateur A. (Paternotte)

Faire une copie du fichier /etc/passwd dans votre répertoire de connexion sous le nom de coucouter

```
paternotte@paternotte-VirtualBox:~$ cp /etc/passwd ./coucouter
paternotte@paternotte-VirtualBox:~$ ls -li
393377 Bureau          393383 Images          393382 Musique        393378 Téléchargements
393398 coucouter       394469 lettres         393380 Public          393384 Vidéos
393381 Documents        393379 Modèles         393241 snap
```

Positionner les droits de ce fichier coucouter à rwx --- ---.

```
paternotte@paternotte-VirtualBox:~$ chmod 700 ./coucouter
paternotte@paternotte-VirtualBox:~$ ls -ld ./coucouter
-rwx----- 1 paternotte paternotte 2947 nov.  3 16:52 ./coucouter
```

Avec les droits classiques Unix, est-il possible d'attribuer à Utilisateur B uniquement le droit r sur coucouter ? Si non, utiliser les ACL Unix.

```
paternotte@paternotte-VirtualBox:~$ setfacl -m u:raoul:r ./coucouter
```

Avec les droits classiques Unix, il n'est pas possible d'accorder à l'utilisateur B uniquement le droit de lecture (r) sur le fichier « coucouter ». En revanche, grâce aux ACL (Listes de Contrôle d'Accès) Unix, l'utilisateur B pourra obtenir uniquement le droit de lecture sur « coucouter ».

Utilisateur B. (raoul)

Visualiser les droits du fichier coucouter. Remarquer le « + ».

```
paternotte@paternotte-VirtualBox:~$ su raoul
Mot de passe :
$ ls -l
total 44
drwxr-xr-x  2 paternotte paternotte 4096 oct.  17 09:50 Bureau
-rwxr-----+ 1 paternotte paternotte 2947 nov.  3 16:52 coucouter
```

Avec la commande `getfacl coucouter`, visualiser son ACL

```
$ getfacl ./coucouter
# file: coucouter
# owner: paternotte
# group: paternotte
user::rwx
user:raoul:r--
group::---
mask::r--
other::---
```

Afficher le contenu de `coucouter`. Possible ? Pourquoi ?

```
$ cat ./coucouter
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

Cela est possible grâce aux ACL Unix, qui permettent à l'utilisateur B d'avoir le droit de lecture (r) sur le fichier « `coucouter` ».

12/ Les redirection

Taper les commandes suivantes et commenter

```
paternotte@paternotte-VirtualBox:~$ who>beurk
paternotte@paternotte-VirtualBox:~$ cat beurk
paternotte seat0          2024-11-03 16:45 (login screen)
paternotte tty2           2024-11-03 16:45 (tty2)
```

La première commande `who > beurk` redirige la sortie de la commande `who` vers un fichier nommé « beurk », créant ainsi un fichier « beurk » qui contient la sortie de la commande `who`. La deuxième commande `cat beurk` permet d'afficher le contenu du fichier « beurk » dans le terminal.

```
paternotte@paternotte-VirtualBox:~$ echo bonjour les amis>beurk
paternotte@paternotte-VirtualBox:~$ cat beurk
bonjour les amis
```

La première commande utilise `echo` pour afficher "bonjour les amis" et rediriger cette sortie avec `>` vers le fichier « beurk ». Par conséquent, le contenu du fichier « beurk » sera "bonjour les amis".

```
paternotte@paternotte-VirtualBox:~$ id>beurk
paternotte@paternotte-VirtualBox:~$ pwd>>beurk
paternotte@paternotte-VirtualBox:~$ cat beurk
uid=1000(paternotte) gid=1000(paternotte) groupes=1000(paternotte),4(adm),24(cdr
om),27(sudo),30(dip),46(plugdev),100(users),114(lpadmin),600(developpeurs)
/home/paternotte
```

La première commande `id > beurk` redirige les informations sur l'utilisateur courant vers un fichier nommé « beurk ». Si le fichier n'existe pas, il est créé ; sinon, son contenu est écrasé. La deuxième commande `pwd >> beurk` ajoute le chemin du répertoire de travail actuel à la fin du fichier « beurk », conservant ainsi les informations précédemment enregistrées. La troisième commande `cat beurk` affiche l'intégralité du contenu du fichier « beurk » dans le terminal, montrant à la fois les détails de l'utilisateur (`id`) et le chemin du répertoire de travail (`pwd`).

Expérimenter les commandes suivantes et conclure :

```
paternotte@paternotte-VirtualBox:~$ ls 1>OK 2>pasOK
paternotte@paternotte-VirtualBox:~$ cat OK
beurk
Bureau
coucouter
Documents
Images
lettres
Modèles
Musique
OK
pasOK
Public
snap
Téléchargements
Vidéos
```

La première commande `ls 1>OK 2>pasOK` effectue les opérations suivantes : `ls` : Exécute la commande qui liste les fichiers et répertoires dans le répertoire courant.
-1>OK : Redirige la sortie standard (stdout) de la commande `ls` vers le fichier "OK". Cela signifie que toutes les informations normales générées par la commande `ls` seront stockées dans le fichier "OK".
-2>pasOK : Redirige la sortie d'erreur standard (stderr) de la commande `ls` vers le fichier "pasOK".

Cependant, comme `ls` ne génère pas de messages d'erreur, le fichier "pasOK" est vide. La deuxième commande `cat OK` affiche le contenu du fichier "OK" sur votre terminal. Cela permet de voir la sortie standard générée par la commande `ls`, qui représente la liste des fichiers et répertoires du répertoire courant. La troisième commande `cat pasOK` affiche le contenu du fichier "pasOK" sur votre terminal. Il est vide car la commande `ls` ne génère pas de message d'erreur.

```
paternotte@paternotte-VirtualBox:~$ cat unfichierbidon 1>OK 2>pasOK
paternotte@paternotte-VirtualBox:~$ cat ok
cat: ok: Aucun fichier ou dossier de ce nom
paternotte@paternotte-VirtualBox:~$ cat OK
paternotte@paternotte-VirtualBox:~$ cat pasOK
cat: unfichierbidon: Aucun fichier ou dossier de ce nom
```

La première commande `cat unfichierbidon 1>OK 2>pasOK` effectue les opérations suivantes :

- `cat unfichierbidon` : tente d'afficher le contenu du fichier fictif « unfichierbidon », ce qui n'est pas possible car le fichier n'existe pas.
- `1>OK` : redirige la sortie standard (stdout) de la commande `cat unfichierbidon` vers le fichier « OK ». Étant donné que « unfichierbidon » n'existe pas, le fichier « OK » restera vide.
- `2>pasOK` : redirige la sortie d'erreur standard (stderr) de la commande `cat unfichierbidon` vers le fichier « pasOK ». Cette redirection capture le message d'erreur généré par la tentative d'affichage du contenu du fichier inexistant.

La deuxième commande `cat OK` affiche le contenu du fichier « OK » sur le terminal. Comme le fichier « unfichierbidon » n'existe pas, « OK » restera vide.

La troisième commande `cat pasOK` affiche le contenu du fichier « pasOK » sur le terminal, où l'on peut voir le message d'erreur indiquant que le fichier « unfichierbidon » n'existe pas.

```
paternotte@paternotte-VirtualBox:~$ (date; cat krzytw; echo fin de commande composée) 1>pre 2>err
paternotte@paternotte-VirtualBox:~$ cat pre
dim. 03 nov. 2024 17:07:22 CET
fin de commande composée
paternotte@paternotte-VirtualBox:~$ cat err
cat: krzytw: Aucun fichier ou dossier de ce nom
```

La première commande (date; cat krzytw; echo fin de commande composée) 1>pre 2>err effectue les opérations suivantes :

- date : affiche la date et l'heure actuelles.
- cat krzytw : tente d'afficher le contenu du fichier fictif « krzytw ». Comme ce fichier n'existe pas, cela génère une erreur.
- echo fin de commande composée : affiche le texte "fin de commande composée".

L'ensemble de ces commandes est encadré par des parenthèses, formant ainsi une commande composée, ce qui permet de regrouper les sorties de chaque commande à l'intérieur.

- 1>pre : redirige la sortie standard (stdout) de l'ensemble de la commande composée vers le fichier « pre ».
- 2>err : redirige la sortie d'erreur standard (stderr) de l'ensemble de la commande composée vers le fichier « err ».

La deuxième commande cat pre affiche le contenu du fichier « pre » dans le terminal, permettant de voir la sortie standard générée par la commande composée, incluant la date et l'heure actuelles ainsi que le texte "fin de commande composée".

La troisième commande cat err affiche le contenu du fichier « err » dans le terminal, où l'on peut voir le message d'erreur concernant le fichier « krzytw » qui n'existe pas.

```
paternotte@paternotte-VirtualBox:~$ ls /bin>/dev/null
paternotte@paternotte-VirtualBox:~$ cat /dev/null
```

La première commande ls /bin >/dev/null effectue les opérations suivantes :

- ls /bin : liste le contenu du répertoire « /bin ».
- >/dev/null : redirige la sortie standard (stdout) de la commande ls vers « /dev/null », un dispositif spécial sous Linux et Unix utilisé pour supprimer la sortie. Ainsi, la liste des fichiers dans « /bin » est éliminée et n'apparaît pas à l'écran.

La deuxième commande `cat /dev/null` tente d'afficher le contenu du fichier spécial « `/dev/null` ». Cependant, comme « `/dev/null` » est un dispositif qui accepte et élimine toutes les données qui lui sont envoyées, il n'a pas de contenu permanent. Par conséquent, cette commande n'affiche rien sur le terminal, car « `/dev/null` » est principalement utilisé pour jeter des données plutôt que pour les afficher.

13/ Les tubes et les filtres

Une petite société de vente de voitures d'occasion gère son parc de véhicules sous Unix. On donne ci-dessous un extrait de la session de travail du responsable de cette société. Le fichier ordinaire `voitures` contient les véhicules en stock disponibles à la vente.

Création du fichier `voitures` avec ses données :

```
paternotte@paternotte-VirtualBox:~$ nano voitures
GNU nano 7.2 voitures *
citroen-XM-1995-diesel-2500
peugeot-604-1978-diesel-1500
citroen-SM-1974-essence-5000
renault-12-1972-essence-500
maserati-quattroporte-1998-essence-36400
citroen-C2-2008-diesel-10200
```

Quel renseignement fournit la commande « `cat voitures | grep ^c` »

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep ^c
citroen-XM-1995-diesel-2500
citroen-SM-1974-essence-5000
citroen-C2-2008-diesel-10200
```

C'est une commande affichant les voitures dont la marque commence par un « c ».

A partir du fichier `voitures` : Donner une commande affichant les voitures électriques.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep électrique
```

Il n'y en a pas.

Donner une commande affichant les voitures essence de marque Peugeot.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep peugeot | grep essence
```

Il n'y en a pas.

Donner une commande indiquant le nombre total de voitures en vente.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | wc -l
6
```

Donner une commande indiquant le nombre de voitures diesel.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep diesel | wc -l
3
```

Donner une commande affichant les voitures qui ne sont pas diesel.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep essence
citroen-SM-1974-essence-5000
renault-12-1972-essence-500
maserati-quattroporte-1998-essence-36400
```

Donner une commande affichant les voitures par ordre croissant.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | sort -n -k5 -t-
renault-12-1972-essence-500
peugeot-604-1978-diesel-1500
citroen-XM-1995-diesel-2500
citroen-SM-1974-essence-5000
citroen-C2-2008-diesel-10200
maserati-quattroporte-1998-essence-36400
```

Donner une commande affichant la voiture la plus ancienne.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | sort -t'-' -k3n | head -n 1
renault-12-1972-essence-500
```

Donner une commande indiquant le prix de la voiture la plus récente.

```
paternotte@paternotte-VirtualBox:~$ cat voitures | sort -t'-' -k3nr | head -n 1 |
cut -d'-' -f5
10200
```

Donner une commande affichant les voitures dont la marque commence par un « p » ou un « c »

```
paternotte@paternotte-VirtualBox:~$ cat voitures | grep -E '^(p|c)'
citroen-XM-1995-diesel-2500
peugeot-604-1978-diesel-1500
citroen-SM-1974-essence-5000
citroen-C2-2008-diesel-10200
```

La société vient de rentrer un véhicule Citroën C6 à moteur diesel six cylindres en V de 2007 dont le prix de mise en vente est fixé à 25 000 euros. Est-il possible d'inscrire ce véhicule dans le fichier voitures sans utiliser un éditeur de texte ? Si oui, donner la commande Unix.

```
paternotte@paternotte-VirtualBox:~$ echo "citroen-C6-2007-diesel-25000">>voitures
paternotte@paternotte-VirtualBox:~$ cat voitures
citroen-XM-1995-diesel-2500
peugeot-604-1978-diesel-1500
citroen-SM-1974-essence-5000
renault-12-1972-essence-500
maserati-quattroporte-1998-essence-36400
citroen-C2-2008-diesel-10200
citroen-C6-2007-diesel-25000
```


14/ L'interpréteur de la commande Bash

En examinant le contenu de la variable d'environnement SHELL, déterminez l'interpréteur de commandes (shell en anglais) que vous utilisez.

Modifier le fichier de configuration personnel.bashrc de telle manière qu'à chaque nouveau shell non de login (connexion en mode graphique ou ouverture d'une consoles de commandes):

- un message de bienvenue de la forme Bonjour s'affiche à l'écran (penser à la variable d'environnement LOGNAME)
- la date s'affiche (commande date)
- la liste des personnes déjà connectées sur la machine UNIX apparaît

```
paternotte@paternotte-VirtualBox:~$ nano ~/.bashrc
paternotte@paternotte-VirtualBox:~$ source ~/.bashrc
bonjour
Date actuelle : dim. 03 nov. 2024 17:32:35 CET
Utilisateur connecté :
paternotte seat0      2024-11-03 16:45 (login screen)
paternotte tty2       2024-11-03 16:45 (tty2)
```

15/ Les variable locale

Définir au niveau de la ligne de commande une variable var contenant la chaîne de caractère "Unix". Attention: il ne faut pas mettre d'espace autour du symbole

Vérifier que la variable var est connue du système grâce à la commande set.

Avec un tube et la commande grep, sélectionner uniquement la ligne concernant var Avec la commande echo, afficher le contenu de la variable var.

```
paternotte@paternotte-VirtualBox:~$ Unix="hello"
```

```
paternotte@paternotte-VirtualBox:~$ set | grep Unix
Unix=hello
```

```
paternotte@paternotte-VirtualBox:~$ echo Unix
Unix
```

Taper l'instruction suivante, destinée à placer dans la variable locale res le résultat du produit de 4 par 7 res-4-7

Que contient la variable res. Remédier au problème en utilisant la notation \$((...)).

```
paternotte@paternotte-VirtualBox:~$ res=4*7
paternotte@paternotte-VirtualBox:~$ echo $((res))
28
```

16/ Les variables d'environnement

Créer un répertoire \$HOME/paf. Dans quel répertoire est créé paf?

```
paternotte@paternotte-VirtualBox:/home$ mkdir paf
```

Paf est dans /home.

A l'aide de la commande whereis, déterminer dans quel répertoire se situe la commande Unix cal (taper whereis cal)

```
paternotte@paternotte-VirtualBox:/home$ whereis cal
cal: /usr/bin/cal /usr/share/man/man1/cal.1.gz
```

Copier cette commande cal dans le répertoire paf sous le nom de calendrier

```
paternotte@paternotte-VirtualBox:/home$ sudo cp $(command -v cal) /home/paf/calendrier
```



Se placer dans /tmp et tenter d'exécuter calendrier sans préciser le chemin, c'est-à-dire en tapant uniquement calendrier. Possible ? Si non, résoudre le problème en modifiant la variable PATH

```
paternotte@paternotte-VirtualBox:/tmp$ calendrier
calendrier : commande introuvable
```

```
paternotte@paternotte-VirtualBox:/tmp$ calendrier
  Novembre 2024
di lu ma me je ve sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

Essayer la commande `cd jojodu80`

Cette commande réussit-elle ? Vérifier en consultant le code de retour (variable spéciale \$?).

```
paternotte@paternotte-VirtualBox:/$ cd jojodu80
bash: cd: jojodu80: Aucun fichier ou dossier de ce nom
paternotte@paternotte-VirtualBox:/$ echo $?
1
```

Essayer la commande `ls/home`. Cette commande réussit-elle ? Vérifier en consultant le code de retour.

```
paternotte@paternotte-VirtualBox:/$ ls /home
paf paternotte pp raoul
paternotte@paternotte-VirtualBox:/$ echo $?
0
```

Interprétation du code de retour

- **0** : Cela signifie que la commande a réussi.
- **1** (ou tout autre code différent de zéro) : Cela signifie que la commande a échoué.

17/ La substitution de commande

Concevoir une commande affichant le nombre de fichiers contenus dans votre répertoire personnel sous la forme: Il y a 24 fichier(s) dans mon répertoire de connexion.

```
paternotte@paternotte-VirtualBox:~$ coun=0; for file in ~/.* ~/*; do [ -f "$file" ] && count=$((count + 1)); done; echo "il y a $count fichier(s) dans mon répertoire de connexion"
il y a 32 fichier(s) dans mon répertoire de connexion
```

Placer dans une variable locale `jour` le nom du jour courant (lundi ou mardi ou...). Tester en affichant à l'écran le contenu de cette variable `jour`

```
paternotte@paternotte-VirtualBox:~$ jour=$(date +%A)
paternotte@paternotte-VirtualBox:~$ echo $jour
dimanche
```

Taper la séquence de commandes suivante, compléter avec les affichages obtenus et indiquer, le cas échéant, quelle substitution est opérée (variable ou commande).

`id "Chuck Norris`

`echo "Bonjour, je suis $(id)`

`echo "Bonjour, je suis id`

`echo "Bonjour, je suis $id"`

```

paternotte@paternotte-VirtualBox:~$ id="Chuck Norris"
paternotte@paternotte-VirtualBox:~$ echo "Bonjour, je suis $(id)"
Bonjour, je suis uid=1000(paternotte) gid=1000(paternotte) groupes=1000(paternotte),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),100(users),114(lpadmin),600(developpeurs)
paternotte@paternotte-VirtualBox:~$ echo "Bonjour, je suis id"
Bonjour, je suis id
paternotte@paternotte-VirtualBox:~$ echo "Bonjour, je suis $id"
Bonjour, je suis Chuck Norris

```

18/ Code de retour d'une commande

Essayer la commande `cd jojodu80`.

Cette commande réussit-elle ? Vérifier en consultant le code de retour (variable spéciale `$?`)

```

paternotte@paternotte-VirtualBox:/$ cd jojodu80
bash: cd: jojodu80: Aucun fichier ou dossier de ce nom
paternotte@paternotte-VirtualBox:/$ echo $?
1

```

Essayer la commande `ls/home`. Cette commande réussit-elle ? Vérifier en consultant le code de retour.

```

paternotte@paternotte-VirtualBox:/$ ls /home
paf paternotte pp raoul
paternotte@paternotte-VirtualBox:/$ echo $?
0

```

Interprétation du code de retour

- **0** : Cela signifie que la commande a réussi.
- **1** (ou tout autre code différent de zéro) : Cela signifie que la commande a échoué.

Exercice sur papier

L'utilisateur Paulo le blaireau tape la séquence de commandes suivantes
`chmod 750 souvenir 2>/dev/null`

`echo $?`

1

Le fichier souvenir appartient-il à Paulo ? justifier succinctement

Le fichier souvenir n'appartient probablement pas à Paulo. En effet, le fait qu'il ait reçu un code d'erreur de 1 après avoir essayé de changer les permissions du fichier suggère qu'il n'a pas les droits nécessaires pour le faire, ce qui est généralement le cas si l'utilisateur n'est pas le propriétaire du fichier.

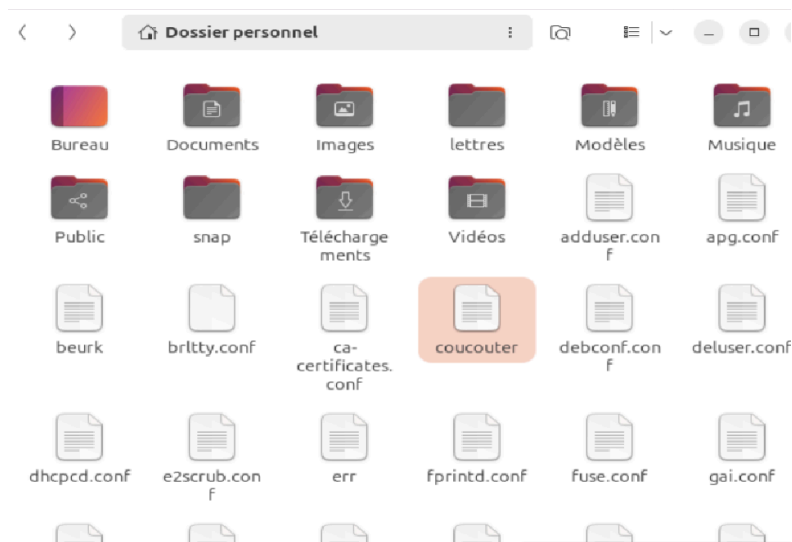
19/ Les métacaractères

Afficher avec la commande `ls` les fichiers du répertoire `/etc` commençant par les caractères `ho`

```
paternotte@paternotte-VirtualBox:~$ ls /etc | grep ^ho
host.conf
hostname
hosts
hosts.allow
hosts.deny
```

Copier les fichiers du répertoire `/etc` à l'extension point `conf` dans votre répertoire personnel

```
paternotte@paternotte-VirtualBox:~$ cp /etc/*.conf ~/
```



Afficher les fichiers du répertoire `/etc` dont le nom comporte trois caractères et commence par un `X` majuscule ou minuscule.

```
paternotte@paternotte-VirtualBox:~$ ls /etc/X?? 2>/dev/null
app-defaults      xinit             Xresources        XvMCConfig
cursors           xkb               Xsession          Xwrapper.config
default-display-manager  xorg.conf.d      Xsession.d
fonts             Xreset           Xsession.options
rgb.txt           Xreset.d         xsm
```

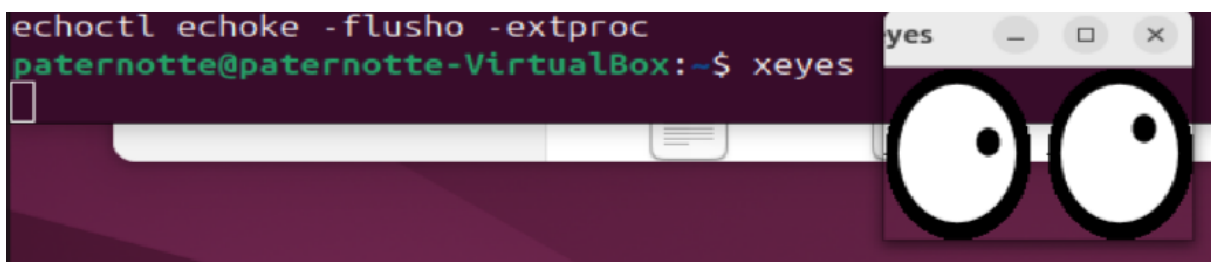
20/ Les processus Unix

A l'aide de la commande `stty-a`, donner les valeurs des caractères `<intr>` et `<quit>`, qui sont respectivement les raccourcis clavier des signaux `SIGINT` et `SIGQUIT`

```
paternotte@paternotte-VirtualBox:~$ stty -a
speed 38400 baud; rows 24; columns 80; line =
intr = ^C; quit = ^\; erase = ^?; kill = ^U;
```

Depuis le terminal courant. Lancer la commande `xeyes` en avant-plan et la tuer depuis le terminal courant.

```
echoctl echoke -flusho -extproc
paternotte@paternotte-VirtualBox:~$ xeyes
```



Lancer la commande `xeyes` en arrière plan, puis tenter de la tuer avec le signal `SIGINT` via le caractère. Possible?

```
paternotte@paternotte-VirtualBox:~$ xeyes &
[1] 5366

paternotte@paternotte-VirtualBox:~$ ps aux | grep xeyes
paterno+  5366  0.0  0.1 20800  4864 pts/0    S   19:27   0:00 xeyes
paterno+  5369  0.0  0.0 17832  2304 pts/0    S+  19:28   0:00 grep --color=
auto xeyes

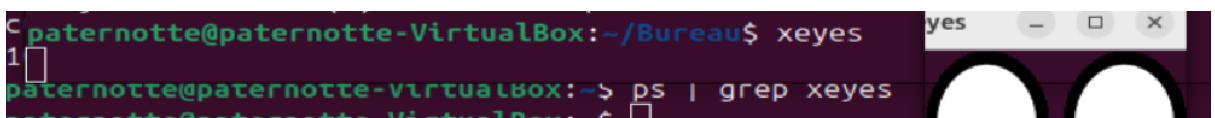
paternotte@paternotte-VirtualBox:~$ kill -SIGINT 5366
```

Depuis un autre terminal. Relancer la commande `xeyes` en avant-plan et la tuer depuis un autre terminal.

Ouvrir une nouvelle console.

Taper la commande `ps` et constater que celle-ci n'affiche que les processus attachés à la console courante (et donc pas `xeyes`...

```
C paternotte@paternotte-VirtualBox:~/Bureau$ xeyes
1
paternotte@paternotte-VirtualBox:~$ ps | grep xeyes
paternotte@paternotte-VirtualBox:~$
```



Utiliser par conséquent la commande ps avec l'option x.

```
paternotte@paternotte-VirtualBox:~$ ps aux | grep xeyes
paterno+  5418  0.0  0.1  20800  4864 pts/1    S+   19:34   0:00 xeyes
paterno+  5422  0.0  0.0  17832  2304 pts/0    S+   19:34   0:00 grep --color=
auto xeyes
[1]+  Fini                  xeyes
```

Taper kill-2 PID.

```
paternotte@paternotte-VirtualBox:~$ kill -2 5418
```

21/ Mise en pause et reprise d'un processus

Lancer la commande xeyes en arrière plan, xeyes& Envoyer au processus correspondant le signal SIGSTOP. Remarquer que le processus est mis en pause (les yeux ne réagissent plus 1).

```
paternotte@paternotte-VirtualBox:~/Bureau$ xeyes &
[1] 6152
paternotte@paternotte-VirtualBox:~/Bureau$ kill -SIGSTOP 6152
paternotte@paternotte-VirtualBox:~/Bureau$
```



Reprendre le processus en lui envoyant le signal SIGCONT

```
paternotte@paternotte-VirtualBox:~/Bureau$ kill -SIGCONT 6152
[1]+  Arrêté                xeyes
paternotte@paternotte-VirtualBox:~/Bureau$
```

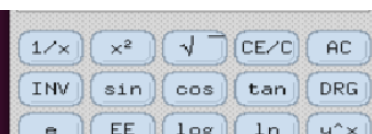


Finalement, tuer le processus en lui envoyant le signal SIGTERM.

```
paternotte@paternotte-VirtualBox:~/Bureau$ kill -SIGTERM 6152
```

Lancer la commande xcalc en avant plan et Mettre en pause cette commande.

```
paternotte@paternotte-VirtualBox:~/Bureau$ xcalc
^Z
[1]+  Arrêté                xcalc
paternotte@paternotte-VirtualBox:~/Bureau$
```



Demander la reprise du processus en arrière plan (commande bg)

```
paternotte@paternotte-VirtualBox:~/Bureau$ bg xcalc
[1]+ xcalc &
```

Re-stopper ce processus.

```
paternotte@paternotte-VirtualBox:~/Bureau$ ps aux | grep xcalc
paterno+  6198  0.0  0.1  21448  5504 pts/0    S    01:57   0:00 xcalc
paterno+  6201  0.0  0.0  17832  2304 pts/0    S+   01:59   0:00 grep --color=
auto xcalc
paternotte@paternotte-VirtualBox:~/Bureau$ kill -SIGSTOP 6198
```


Le reprendre en avant plan (commande fg)

```
paternotte@paternotte-VirtualBox:~/Bureau$ fg xcalc
xcalc
```

Le re-stopper et Finalement, tuer le processus avec le signal SIGKILL.

```
paternotte@paternotte-VirtualBox:~/Bureau$ fg xcalc
xcalc
^Z
[1]+  Arrêté                  xcalc
paternotte@paternotte-VirtualBox:~/Bureau$ kill -SIGKILL 6198
```

22/ Les scripts-shell

■Script-shell: programme écrit à base de commandes Unix

nano le nom du script-shell avec extension sh, puis entrée, vous allez crée votre premmier script

Le droit x doit être positionné sur le script shell

chmod u+x script.sh

```
paternotte@paternotte-VirtualBox:~/Bureau$ nano script.sh
paternotte@paternotte-VirtualBox:~/Bureau$ chmod u+x script.sh
```

■Toute ligne commençant par un # désigne un commentaire

- Sauf la première ligne qui doit être obligatoirement:

#!/bin/bash

Ceci est un commentaire

```
GNU nano 7.2 script.sh
#!/bin/bash
#Ceci est un commentaire
```

23/ Conclusion

Dans ce SAE, j'ai appris un nouveau langage de commande : Linux. Bien que j'aie été absent pendant plusieurs séances, j'ai réussi à rattraper mon retard chez moi. Ce projet m'a permis de comprendre le fonctionnement de Linux, ainsi que les concepts fondamentaux liés à la gestion des fichiers, aux permissions, et à l'utilisation de diverses commandes. J'ai également développé des compétences pratiques en matière de redirection de sortie, de manipulation de fichiers, et d'utilisation des outils en ligne de commande. Cette expérience m'a non seulement enrichi sur le plan technique, mais m'a également encouragé à approfondir mes connaissances en informatique. Je suis maintenant plus confiant dans mes capacités à travailler dans un environnement Linux.